ISCR

Institute for Scientific Computing Research

Student Internship Research Summaries

CASC Center for Applied Scientific Computing

Out-of-Core KD-Partitioning of Large-Scale Data Sets for Distributed Volume Rendering

Janine Bennett
University of California, Davis

Summary:

There is need for new ways to visualize large-scale data sets. Current methods can be very slow and inefficient since the data sets are larger than can fit in core memory. By dividing up the data sets into smaller load-balanced parcels, and by distributing them over a cluster, we can speed up visualization times dramatically.

Because of memory constraints, an out-of-core kd-partitioning algorithm is used. The vertices of the data set are partitioned spatially into three-dimensional buckets. Further partitioning is then done based on these counts rather than on the actual vertices themselves. If the granularity of the buckets is insufficient to achieve a good balance, the buckets are subdivided recursively until a good balance is achieved. The data in each of the final partitions is stored on disk as an Mpack block. In addition, MPI code was written to transfer these Mpack blocks between nodes on a cluster; this is used to enable load-balancing. Peter Williams, Nelson Max, and Mark Duchaineau were all involved in different stages of the project.

This code will now be used to partition large data sets into smaller blocks which can each be volume rendered in parallel on a node of a cluster.

Parallel Isosurface Refinement

Rita Borgo
University of Pisa, Italy

Summary:

Multi-resolution data-structures and algorithms are key techniques in Scientific Visualization used to achieve real-time interaction with very large data sets. Most research has been primarily focused on the off-line construction of multi-resolution representations, mostly using decimation schemes. Drawbacks of this class of approaches include the inability of maintaining interactivity when the displayed surfaces change frequently. Moreover, it is difficult to guarantee the consistency of the geometric embedding (no self-intersections) of any approximated level of detail of the output surfaces. In our project we have been working on a solution to such problems restricting our attention to the case of meshes computed as isocountours of 3D scalar fields. We adopt a multi-resolution data-structure for the output surface that allows trading accuracy for speed, thereby achieving interactive response times in graphical display. The main feature of our scheme is the ability to extract in a multiresolution fashion the input volumetric data to build a multiresolution version of the output by successive refinements. We use a refinement primitive that updates a local portion of the output so that a consistent output mesh is maintained at any given time. This allows for asynchronous termination of the computation within a small constant delay.

To realize our progressive refinement algorithm and data-structure we use the edge-bisection refinement scheme, which is widely used in the meshing community. This scheme allows one to build a multiresolution data-structure for regular grids without preprocessing since it is equivalent to a predetermined order (octree-like) of reading the vertices. The 3D mesh partitions the region of space of interest into tetrahedra. Each vertex is associated with an input function value. Inside each tetrahedral cell a linear function is used to interpolate the function values at the vertices. In this way we have a piecewise linear representation of the scalar field necessary to compute an isocontour. The coarse level of our refinement is represented by tetrahedra; each refinement step inserts a new vertex on an appropriately chosen edge, splitting each tetrahedron incident on such edge in two halves. The isosurface is updated within each tetrahedron. In this way, as the edge-bisection algorithm makes progress, new function values are introduced and a more detailed definition of the function is obtained. Instead of recomputing portions of the contour, we augment its representation generating directly a progressive data-structure. The refinement can be applied locally to perform adaptive refinement or globally to increase uniformly the resolution of the mesh. This progressive representation of time isocontour can be traversed adaptively independently from the underlying mesh. This kind of approach enables a new kind of tradeoff between speed and accuracy. In our prototype implementation we uncoupled the isosurface construction from its display. The isocontour hierarchy is built by one process that traverses the input 3D mesh. A second process that never accesses the 3D mesh performs the isocontour traversal arid display. For the display part we take advantage of a parallel computation scheme partitiong the data in an efficient tradeoff between load balancing and timing constraints.

We are currently working on the design of alternative traversal strategies of the tree-like structure to improve efficiency of the refinement and to provide a consistent construction of the output mesh in parallel and with minimal interprocess communication. In particular the goal is to design the isosurface construction and a one-way stream of data so that it can work reliably, with no need of synchronization.

Topology-Preserving Subdivision

Peer-Timo Bremer
University of California, Davis

Summary:

In the field of scientific visualization, the computation of isocontours remains one of the most important tools. Of special significance is the topology of isocontours. During visualization no artificial topology should be created and no existing topology should be destroyed. During direct isocontour extraction, these requirements are fulfilled if a sufficiently small numerical error bound can be guaranteed. However, many common data sets become too large for a direct isocontour extraction, especially for real-time applications. The most common solution to this problem is to compute a multiresolution hierarchy in the form of a wavelet transform of the data. The hierarchy can then be used to speed up the isocontour extraction. However, wavelet transforms also introduce artificial topology, especially in the lower resolution representations of the data. For example, new minima and maxima are created much like the over- and under-shoots known from spline approximations. This new topology distorts the real data and can mislead scientists. Our goal is to find a subdivision scheme that allows us to have large differences in resolution while maintaining the correct topology.

To explore the possibilities of topology-preserving and topology-controlled subdivision, we use two-dimensional height fields. As the underlying subdivision scheme we chose the "4-8 subdivision" as described by Velho and Zorin. The 4-8 subdivision is based on semi-regular meshes and bisection refinement. The advantages are the comparatively slow growth factor in each subdivision step and the high-resolution differences possible without creating cracks. However, there has been no method of computing the wavelet coefficients for an adaptively refined 4-8 mesh. Even though the masks that are used during subdivision are small, the evaluation requires a nearly uniformly subdivided mesh. We have developed a new lazy evaluation scheme based on guessing necessary values and correcting possible errors later. Our method allows us to use a fully adaptive 4-8 mesh that uses approximated data at places where data is missing. If the correct data is supplied (a necessary value inserted) the resulting change is distributed over the mesh to correct the difference between the former guess and the real value. If all necessary information is introduced, our mesh is the same as one created by uniform subdivision.

Using the above technique we are developing a subdivision scheme where new topology is inserted only at specific user-defined points. The insertion process is guided by a hierarchy of Morse complexes that describe the topology of the data for different error bounds. The mesh is controlled by the Morse hierarchy in the sense that only the topology of the corresponding Morse graph exists in the mesh. For isocontour extraction this means that no isocontour is lost and no artificial isocontour is created. We avoid creating additional topology by enforcing the following rule: No vertex in the mesh that is not part of the Morse graph (is not an explicitly noted critical point) is allowed to be a critical point. This condition can be checked by visiting the neighborhood of a vertex. However, enforcing the condition without distorting the mesh too much is a challenging problem.

Future research will focus on combining the current adaptive refinement with a full implementation of the topology-preserving algorithms and on methods to control the topological changes. The long-term goal is to lift the scheme to three dimensions.

A New Parallel Coarsening Method for Algebraic Multigrid

Oliver Broeker
Swiss Federal Institute of Technology

Summary:

Algebraic multigrid (AMG) is known to solve a large class of linear equations arising from the discretization of partial differential equations. Part of the algorithm is a greedy procedure to choose the coarse unknowns for a given system. This procedure is inherently sequential. Parallel implementations of an AMG typically use an approach where sequential coarse grid choice is done on the local grids and boundary points are treated separately. The resulting convergence rate is dependent on the number of processors. We seek a parallel coarsening process that is free of this limitation.

We devised an algorithm based on ideas similar to the "compatible relaxation" of Brandt. A Matlab implementation yields good convergence rates and bounded operator complexities for a variety of small anisotropic diffusion and convection diffusion problems. The algorithm was added to the BoomerAMG code and tested on larger matrices.

The method needs further investigation, especially since the choice of its threshold parameters is crucial and nontrivial. The work will be continued at the ETH as the focus of a doctoral dissertation.

Using Meta-Data to Automatically Wrap Bioinformatics Sources

David Buttler
Georgia Tech

Summary:

A large amount of bioinformatics data is distributed over the Internet. Typically, this information is accessible only through custom, web-based query interfaces. These interfaces often include features uncommon in industrial applications such as multiple parameters, a variety of query options that invoke different support programs, indirection and delay pages, and complex results. If users require information from multiple sources, they must pose the appropriate queries at each source individually then explicitly integrate the results. This solution may be acceptable for a small number of sources, but it quickly becomes an overwhelming burden for users. Currently there are over 500 bioinformatics sources for biologists to choose from, making it infeasible to manually gather data from a non-trivial fraction of the available sources.

Our goal is to simplify access to bioinformatics data by providing a single access point to a large number of sources. The fundamental problem is that each source is fiercely independent. As a result, they use different semantics, customized interfaces, and unique data formats. Furthermore, they are prone to having their interfaces and formats frequently updated without warning. In order to maintain access to a source, a specialized access program (wrapper) that keeps pace with the source's evolution is required. We present a general meta-data format capable of describing the complex, web-based interfaces often found in bioinformatics. Because this description provides sufficient information to automatically generate a wrapper for the associated site, it is an important first step in developing an infrastructure capable of interacting with a large number of dynamic, heterogeneous data sources.

Rather than create a new meta-data format from scratch, we started with the DAML specification. DAML is an extension of XML and RDF that allows the definition of machine understandable ontologies. Specifically, we have extended DAML's web service description language, DAML-S. A service in DAML-S is essentially a web-based interface to data. The DAML-S description of a service has three parts: a service presents a profile, which indicates what information the service requires (input) and what the service does (output); it is described by a service model, which indicates how each step of the service works; and it supports a service grounding, which indicates how to access the service. For the purposes of our data source description, a profile is the external interface to a web service, the model represents the components of the service, and the grounding is the detailed information about how to properly interact with the service interface.

The DAML-S model describes how components of a service interact. The most important characteristic of this description is the Process, which has inputs, outputs, and effects (real world consequences for invocation). Processes can represent atomic actions, or control flows (e.g., looped, conditional, sequenced [ordered], and parallel [unordered] execution). To complete the interface descriptions of data sources, we have extended DAML-S to include a detailed specification for groundings (the current release does not define them). Our definition of a grounding contains several elements: (1) a pointer to the Process it grounds; (2) the binding; (3) a specification of how to convert Process inputs into parameters the binding can use; (4) a parser; and (5) links.

Summary (continued):

David Buttler
Georgia Tech

We hope that this work will speed the evolution of the DAML-S standard and help establish a common format for complex interface descriptions. However, our primary motivation for this work has been to make the first steps towards an infrastructure that can support access to a large number of dynamic, heterogeneous scientific data sources. As such, we are continuing to develop this infrastructure by extending the XWrap Elite wrapper generator to use our metadata descriptions, instead of human interaction, to generate a wrapper for the associated source. Once completed this program will generate full-featured wrappers that conform to a common query interface and return objects in XML. Providing this uniform source interface will enable different integration scenarios, from simple multi-source result fusion to techniques that reconcile semantic information such as data warehouses or federated systems, to be explored. Thus, it is an important first step in providing a single access point for bioinformatics researchers.

A Generic Scheduling Simulator for High-Performance Parallel Computers

Gyu Sang Choi
Penn State University

Summary:

Job scheduling is very important to meet versatile user demands in large-scale, high-performance computing environments. Since a systems environment is constantly changing, redesign or reevaluation of scheduling policies is frequently needed. Generally, system designers choose the simulation approach to evaluate different scheduling policies. The development of a new scheduling simulator, however, is a time-consuming process. Nevertheless, the simulator cannot be reused easily for a new environment. Thus, it is very crucial to develop a scheduling simulator independent of environments.

We propose a generic scheduling simulator for high-performance parallel computers. The simulator is based on an M-square framework for representing various systems. We expand the M-square framework because it has some limitations for a generic scheduling simulator. The proposed simulator supports various homogeneous machine architectures, takes various workloads, makes a user describe easily a new scheduling policy, and calculates memory and context switching overhead. Therefore, a system designer can avoid the overhead to develop a simulator, using the simulator.

There are some limitations in a current general scheduling simulator. This simulator does not model dynamic events like message passing and synchronization mechanisms, mainly due to the lack of a mechanism to represent inter-task communication patterns. We will expand this simulator to support dynamic events in order to run the simulation for implicit co-scheduling or dynamic co-scheduling. Currently, the simulator supports only a homogeneous environment, but we plan to support heterogeneous environments in the next version. The generic simulator can measure a memory overhead but does not run the simulation with a network model. Since this issue is related to message passing and synchronization mechanism, these two issues will be manipulated together in the future.

Two-Grid Methods for Radiative Transfer

Todd Coffey
North Carolina State University

Summary:

Nonlinear radiation diffusion problems can be notoriously difficult to solve implicitly. Application of nonlinear solvers requires repeated solution of a linearized version of the problem, and these linear problems are often the bottleneck for the entire solution process. We are trying to improve on these traditional methods by developing a two-grid scheme in which the problem is solved very accurately on a coarse mesh followed by a single solve of a linearized version on the original problem mesh. The motivation for the method is that the coarse-mesh solution will capture the nonlinear behavior of the problem. Thus, when the solution is moved to the fine-mesh we need only refine it with a linear solve. As a result the nonlinearities are dealt with on a much smaller mesh than the original fine mesh, resulting in a faster solution process. This method was successfully applied to a similar problem arising in ground water applications by Wu and Allen in 1999. The main theoretical development for this method comes from J. Xu (*SIAM Journal of Numerical Analysis*, 1996) and the extensions to our nonlinear system come from C.N. Dawson, M.F. Wheeler, and C.S. Woodward (*SIAM Journal of Numerical Analysis*, 1998).

We implemented this method by modifying an existing code for solving this problem. The existing code employs LLNL's PVODE as an adaptive time-stepping ODE solver (BDF time integrator and Newton nonlinear solver), transport3d (a radiative transfer in-house development code) providing the right-hand side function along with *hypre* (LLNL's high-performance preconditioner library) helping with parallelization and the preconditioner. We modified PVODE to handle two grids and added that capability to transport3d. With these changes, we could then solve the coarse mesh problem using the same machinery as before. The grid transfer routine was then written using trilinear interpolation on the transport3d side. Finally, a fine-grid linear solve will be written that will use the preconditioned GMRES solver from PVODE and a new right-hand side function from transport3d. The changes to both codes abstract the method so that it can be applied to different problems. Eventually this method may migrate to KINSOL and IDA (LLNL's nonlinear system solver and an implicit DAE system solver).

We expect this two-grid method to be especially effective in very nonlinear problems where the nonlinear solver in PVODE requires many iterations. There are several physical parameters in the radiative transfer problem that we can adjust to make the problem more nonlinear. We will conduct a series of experiments to determine how this method compares with the existing solution method.

Computations in Scalar Field Topology

Kree Cole-McLaughlin
University of Utah

Summary:

Scalar fields are used to represent data in different application areas like geographic information systems, medical imaging, or scientific visualization. One fundamental visualization technique for scalar fields is the display of isocontours, that is, sets of points of equal scalar value, also called the isovalue. For example, in terrain models isolines are used to highlight regions of equal elevation. The Contour Tree is a graph that represents the relations between the connected components of the isocontours in a scalar field. Two connected components that merge together (as one changes continuously the isovalue) are represented as two arcs that join in a node of the graph. Each node occurs at a critical point in the scalar field. For a simphicial domain with a piecewise linear interpolant, Carr et al. present a simple algorithm for computing the Contour Tree with complexity O(nlogn+ N), where n is the number of points in the domain, and N is the number of simplicies.

One fundamental limitation of the Contour Tree is the lack of additional information regarding the topology of the contours. The topology of an isocontour is fully characterized by its Betti numbers. In 3D fields the isocontours are surfaces and the Betti numbers correspond to the number of connected components, the number of tunnels, and the number of voids enclosed by a surface. Pascucci has presented a simple and fast algorithm for the construction (if the Contour Tree of a 3D scalar field augmented with the Betti numbers of each contour). The complexity of this approach is O(N log N) in time and O(N) in storage. The nodes of the Augmented Contour Tree correspond precisely to every critical point of the field. Thus, we trade a slightly slower algorithm for more information.

We have implemented both of the above algorithms and produced images of complex contour trees for a few datasets. The images are produced with the graphviz utility from AT&T. Our datasets are quickly approaching the limit of this utility's ability. Therefore, in the future we will need to develop our own tool for drawing the Contour Tree. We have designed and implemented a third algorithm, which has not been published yet, that computes the Augmented Contour Tree in O(n log r+N) time. This new algorithm combines the efficiency and simplicity of the Carr scheme in with the augmented information introduced in the Pascucci scheme. Thus we have successfully introduced new and useful information to the Contour Tree data structure without slowing down the computation.

We have several goals for future development of this project. Currently we are working on scaling the computation up to very large datasets by working out an algorithm that can be implemented on parallel computers. This new approach has an added advantage in that it does not require a simplicial domain or a piecewise linear interpolant. Hence it is applicable to the trilinear interpolant that is commonly used for rectilinear grids. We are working on improving the complexity of the current algorithm by eliminating the O(n log n) term. There are plans to implement the Contour Tree as an interface to visualization software. This opens up a whole range of possible information that can be presented for each contour. We are also working on the problem of drawing the Contour Tree for time-varying fields.

Parallelization of the HIAC Volume Renderer and Efficient Determination of an Exact Visibility Ordering

Richard Cook
University of California, Davis

Summary:

The HIAC volume renderer is software that does volume rendering of unstructured grids. The zoo cells formed by an unstructured grid are first sorted into a visibility order and then projected onto the viewplane using an algorithm by Nelson Max and displayed using OpenGL hardware rendering.

The projection step is the slow step in the program. Therefore, in this project the projection and rendering steps were to be parallelized to improve performance on large datasets.

The parallelization of the projection step was done using pthreads under a shared memory processor architecture. The work process is pipelined into sort, projection, and OpenGL steps, with multiple projecting threads, as that step is the bottleneck. Each projecting thread is added to a queue of consumers and one or more threads at the head of the queue are awakened each time the producer finishes a quantum of work. As each finishes, a final rendering thread converts the information stored by the projection threads directly into optimized OpenGL hardware calls. Near linear speedup was obtained for limited numbers of threads, but scalability in the sense of constant speed for increasing data set sizes has not been demonstrated.

During the parallelization effort, we also created and implemented a replacement for the MPVO sort by Peter Williams. The new Scanning Exact MPVO (SXMPVO) algorithm performs an exact (to the resolution of the resulting image) visibility ordering of the zoo cells and is very fast.

The parallelization and visibility ordering work was significant enough to result in a Master's thesis for Richard Cook and two paper submissions to the *IEEE Visualization Conference* in 2001.

HIAC can be integrated into a user tool that is scalable and robust to be used by scientists at the Laboratory to generate high-accuracy images of their data. This may require re-engineering HIAC's research code using a more modular and extensible approach.

Overlapping Finite Elements for 2D Linear Elastic Problems

Nathan Crane

University of Illinois, Urbana-Champaign

Summary:

Standard finite element analysis is performed on a fully compatible mesh. This implies that inside a continuum body all nodes occur only at the corners of an element and that there are no gaps or overlaps between elements. CAD packages generally do not construct a compatible model. Instead a CAD package produces a solid from the union, intersection, or subtraction of simple parts. Each of these simple parts such as cylinders, NURBS surfaces, and trimming curves are individually very easy to mesh. However, producing a compatible mesh from these intersecting simple parts is very difficult and often requires extensive human intervention. An alternative mesh construction strategy is to not bother producing a compatible mesh but instead solve the problem on a set of overlapping meshes.

We produced a scheme that creates a valid 2D mesh from overlapping elements. This method is implemented by creation and constraint of boundary nodes. This scheme is compatible within the framework of a standard iterative solver. Correctly formulated overlapping meshes will pass the "patch test." Thus, with sufficient refinement the exact solution to a problem can be obtained. Experiments on complex geometries demonstrate that overlapping meshes are as accurate as a corresponding compatible quad mesh and much superior to a triangle element-only mesh. Overlapping meshes do require more elements and more degrees of freedom than a corresponding compatible mesh. Though a overlapping mesh may be computationally slightly less efficient, it can be constructed with less human interaction, reducing total problem solution time and effort.

Parallel Gauss–Seidel Methods

Paul Dostert
Texas A&M University

Summary:

We are interested in numerical solutions to the general sparse matrix problem, $Ax=b$. Typically, A comes from some finite element discretization of an elliptic PDE. A Gauss–Seidel algorithm is frequently used as a smoother or preconditioner for many numerical techniques which solve $Ax=b$. Often, general solution methods are applied on extremely large sparse matrices, making parallel algorithms a necessity. Unfortunately, the Gauss–Seidel algorithm is not easily parallelized because each point in a grid needs recently updated values of some previous points. We investigate different distributed memory parallelizations of a Gauss–Seidel algorithm for unstructured grids.

In order to begin any type of parallelization for Gauss–Seidel, we must first use a coloring technique. The idea of coloring is that we may run different pieces of a grid that never communicate directly with one another, simultaneously. Two different coloring techniques were used for our algorithm: nodal coloring and domain coloring. Nodal coloring involves assigning each node in our domain a color such that no two neighboring nodes have identical colors. We then run through the colors sequentially, but may distribute the nodes of any given color over many processors. One problem is that since the colors are spread out throughout our domain, passing these individual nodes between processors is a time-consuming task. This prompted us to look at domain coloring.

To perform domain coloring we must first decompose our grid into multiple non-overlapping domains. This is accomplished using MeTiS, an element-based splitting program. Differences between two MeTiS versions, pmetis and kmetis, were also analyzed. Once we split our domain, we use the same technique as with the nodal coloring, only we color whole domains now. Therefore, we color the domains such that no two neighboring domains have the same color. One large advantage over nodal coloring is that only the elements that are in multiple domains must be passed.

All of our techniques were programmed in C using MPI. The algorithms, for large problems, were run on the Compass Clusters and ASCI Blue Pacific. The reduction of passing from nodal to domain coloring increased the speed of the Gauss–Seidel almost ten-fold for some domains. We were also able to obtain speed increases as we used more processors for the domain coloring code.

Possibilities for further work include switching from a traditional Gauss–Seidel algorithm to one with overlap (Schwarz). This will involve decomposing our grid using MeTiS, then creating an overlap between the domains. Once this overlap is made, the new domains with overlap will be colored, and then run in parallel. A serial overlap algorithm converges in approximately half the iterations of a traditional serial Gauss–Seidel. Our hope is that the increase in passing and work done per iteration will be offset by an increase in convergence rate, therefore decreasing overall computation time.

Optimal Shape Design for a Layered Periodic Helmholtz Problem

Mike Flanagan
Texas A&M University

Summary:

A standard model of the Helmholtz equation was studied, using non-regular elements applied to an h-p refinement mesh. One of the problems inherent in the discretization of the Helmholtz equation is unwanted reflections in the outer boundary. An attempt is made to resolve the function to its natural analytic state as we move towards the outer boundary. The idea was to use a simple geometry for the domain and develop a triangulation (here, rectangles were used with tensor product basis functions) scheme that would allow for the p-refinement to increase in order as we move further from an internal boundary, being a square in this case. Also, the mesh size increases as we move out, with the essential rule being that two elements' faces form a side of the next face as we move out in layers from the internal boundary. The boundary conditions employed at the far end are standard Sommerfeld Radiation conditions. We have developed a two different mesh schemes, and further work will include studying the associated mass-matrices resulting from such a scheme. Issues to be resolved are potential numerical instabilites associated from the larger degree polynomials.

Low Machnumber Compressible Flow Simulations Using a Multigrid Procedure

Achim Gordner
University of Heidelberg

Summary:

Compressible flow simulations in a low Mach number regime suffer under the existence of acoustic waves. In particular, long wavelength pressure fluctuations are coupled with short wavelength velocity fluctuations. Since it is our intention to perform aeroacoustic flow simulations, the accurate resolution of the above-mentioned coupling is essential for an appropriate simulation. Having this multiscale coupling problem in mind, a multigrid solution procedure is chosen. However, standard multigrid procedures cannot be applied, since the resulting system of algebraic equations tends to become stiff if the Mach number goes to zero. We concentrate on modifications introduced into the smoother to make it applicable to resolution of the pressure-velocity coupling.

Recent approaches include using the multiple pressure variable technique. Since therein a certain decoupling between the pressure and velocity fluctuations is mandatory, we decided to use a full approach using the entire compressible Navier–Stokes equations for an ideal gas. We extended the primitive variable discretization based on our experience for incompressible flow simulations to the compressible case. Since off-diagonal matrix entries are dominant and, therefore, should be taken into account within the smoothing process, a strong incomplete LU factorization was chosen to act as smoother within the multigrid cycle. Convergence of the smoother was reached for low Mach numbers with appropriate numbering of unknowns and introduction of some extra fill-in. However, developing an optimal algebraic scheme limiting necessary fill-in is still under way. To prove that the implementation of the incomplete LU factorization is still applicable in parallel computations for this application, we employed the CASC Compaq cluster.

Modifing the smoother within the multigrid cycle to count for the multiscale coupling is one possibility for adapting the multigrid solution procedure. Another solution may be the proper choice of restriction and prolongation operators.

Interactive Exploration of Large Iso-Surfaces in Volume Datasets

Benjamin Gregorski
University of California, Davis

Summary:

Numerical simulations performed on LLNL supercomputers are generating unprecedented large amounts of data. A standard method for visualizing, exploring, and understanding this data is to examine various isocontours. The problem with directly viewing the isocontours is that they contain hundreds of millions of triangles. Surfaces of this size cannot be viewed at interactive frame rates on conventional desktop machines. Intelligent preprocessing techniques and runtime algorithms are needed for interactive exploration and visualization. The goal of this project is to develop a system for dataset exploration on desktop workstations. Large datasets are divided into subsets called bricks and stored on disk or on a remote system. These bricks are loaded and unloaded by the application as they are needed so that system resources are effectively utilized.

Interactive frame rates can be achieved using algorithms that employ multiresolution data structures with view-dependent rendering. Multiresolution data structures make better use of available storage space and computation power by representing important areas with more detail and less important areas with less information. View-dependent rendering selects what portion of the data to render based on where the user is looking and on the user's perception of the data. Objects outside the field of view do not need to be rendered, and objects that are far away can be rendered at lower resolutions. View-dependent rendering takes advantage of the multiresolution data structure to efficiently select what to draw.

Our multiresolution data structure is a recursive tetrahedral mesh based on longest edge bisection. The mesh structures supports fast, local refinement necessary for view-dependent rendering. In addition, it supports an efficient method for ensuring mesh continuity required for iso-surface extraction. View dependent rendering calculates the distortion on the view screen; it adjusts the refinement of the multiresolution mesh, selecting finer levels where more detail is needed and coarser levels where less detail is needed.

The basics of the multiresolution data structure and view-dependent refinement have been implemented. Our continuing work is focused on working with large datasets, accurate representation of the data, and computational optimizations.

The Fast Multipole Method for Computational Electromagnetics

Chaz Hales
Brigham Young University

Summary:

Integral equation methods, such as the Method of Moments, are useful in computational electromagnetics because they accurately model the electric or magnetic field resulting from a given source distribution. Because the field resulting from a point source is well known, we can easily find the field from a source of arbitrary distribution by integrating this "point source" expression over the desired source distribution. The resulting integral equation is in fact a linear system of equations and can be solved in a straightforward manner. An important drawback in solving these linear systems of equations is the fact that the matrix describing the interactions between these source points is a large, full matrix. The time required to find the solution for such a system is on the order of *N*-cubed for direct solution methods, where *N* is the number of points, or unknowns. Even faster iterative solution methods require *N*-squared computations for each iteration. As we desire to solve larger and larger problems (that is, as the number of unknowns *N* increases), we are hindered by this computational bottleneck.

The Fast Multipole Method (FMM) yields computational savings by decreasing the number of computations required per iteration of the solution method. At the core of FMM is the idea that direct interactions between one point and every other point can be adequately accounted for by first grouping the points into smaller clusters. Rather than computing the interactions between all the points directly, only the interactions between these groups are computed, thus reducing the number of computations required to solve the system.

Currently we have implemented FMM for a very simple electromagnetic problem. We have used FMM to solve the linear system resulting from a perfectly conducting (PEC) sphere illuminated by a simple planewave in free space. This canonical problem serves as a benchmark that allows us to compare initial FMM results to an exact analytical solution. Accuracy of three decimal places was achieved in test cases, and it is hoped that additional accuracy can be achieved in future test runs.

Immediate plans for the future include extending this simple PEC case to allow us to model other materials (dielectrics). Also a Multi-Level Fast Multipole Algorithm (MLFMA) will be implemented allowing us to achieve even greater computational savings. The net effect of these changes will allow us to solve a broader category of problems, as well as solve larger problems than currently can be solved in reasonable run time.

Adaptive Mesh Refinement for Oceanic Carbon Sequestration Simulations

Aaron Herrnstein
University of California, Davis

Summary:

The continuously increasing rate of CO_2 emissions into the atmosphere is an ever-growing concern of climatologists. It is hypothesized that an excess of such gases will produce a "greenhouse" effect, thus increasing global temperatures and ultimately altering global climate. As an alternative to releasing greenhouse gases into the atmosphere, it has been proposed to deposit them in the ocean. Major concerns of this carbon sequestration are the amount of CO_2 that escapes back into the atmosphere and biological impacts brought about by possible changes in ocean pH.

Ocean General Circulation Models (OGDM) are used to model sequestered carbon over a time scale of centuries. The OGCM used by LLNL's Climate and Carbon Cycle Modeling group shows discrepancies in pH changes between fine and coarse grids. A finer grid is needed to determine convergence, but refining the entire grid will be quite costly in terms of computer time (on the order of months in running time). Adaptive Mesh Refinement (AMR) allows desired sections of the grid to be refined, thus reducing run time considerably.

It is the goal of this research to produce an OGCM that uses AMR. Such a tool will be very useful for areas such as Carbon Sequestration. To aid AMR implementation, the software package SAMRAI (Structured Adaptive Mesh Refinement Application Infrastructure) is used. Refinement of the tracer field is based on gradients and possibly other criteria as needed. In addition, regions of critical topography might be refined if necessary. Current capabilities include a fully functional tracer advection model using AMR and a diagnostic meshing code that uses data from a traditional OGCM. Work for the immediate future involves creation of a shallow water dynamics model using AMR.

Euclid Parallel Preconditioning Library

David Hysom
Old Dominion University

Summary:

Euclid is a scalable parallel ILU preconditioning library that supports arbitrary levels of fill. The implementation is based on our PILU algorithm that was presented at SuperComputing 1999, and later published in expanded form in the *SIAM Journal on Scientific Computing*. This summer brought the work to fruition, with many experimental studies conducted on the ASCI Blue platform, and the code's inclusion in the LLNL *hypre* library.

The algorithm takes a matrix as input and assumes no *a priori* knowledge of structure, symmetry, or other unique features. The only requirement for good performance is that the graph of the matrix be partitionable such that each subdomain has a large interior to boundary node ratio. This requirement is met in practice by most mesh and problem generation codes in the physical sciences.

This summer's experiments centered around two key questions. First, how does the preconditioning technique scale when problem size is increased in proportion to the available processing power; second, how does PILU performance compare to Block Jacobi preconditioning (Block Jacobi being a well-known preconditioning technique that is trivially parallelizable but not scalable in convergence rate). The following representative experimental results are for the IBM-SP Blue Pacific machine at LLNL.

The triangular solve (a.k.a., preconditioner application) phase scales on a per iteration basis, as problem size is scaled in proportion to the number of processors. In one of our examples, a 2D convection-diffusion problem, each processor owned a grid of approximately 65 thousand unknowns. Global problem size was increased from 65 thousand unknowns (one processor) to 26 million unknowns (400 processors). We tested factorizations with various fill levels, e.g., ILU(1), ILU(3), and ILU(6). In all three cases the lines "ramp up" in the range of 1 to small numbers of processors. This is expected and is attributable to "filling up" the communication pipeline. The near flatness of the execution time performance for increasing problem size from 64 through 400 processors for all levels indicates near perfect scalability.

Tests on a simplified 2D nonlinear radiative transport problem on up to approximately 4 million unknowns on 400 processors compare our new PILU preconditioning technique with the well-known and highly parallel Block Jacobi technique. Results show that PILU preconditioning is increasingly advantageous as problem size increases. For 400 processors, PILU preconditioning cut total solution time by 50%. PILU is not convergence-rate scalable as is, for instance, multigrid. However, it is a very strong preconditioner with excellent per-iteration scaling properties. Hence, it could be embedded in a multilevel algorithm as a smoother.

Using Genetic Algorithms for Image Processing

Ty Jones
University of Nevada, Reno

Summary:

The Sapphire group is working toward an automated image understanding system. Image processing is a very important part of image understanding. To automate such a system, the use of genetic algorithms is potentially very useful. We decided to try implementing the Phoenix segmentation algorithm with genetic algorithms. This approach was attempted first by Bir Bhanu with promising results.

The first step was to implement the Phoenix algorithm. The algorithm utilizes the histogram of the image to divide the image into smaller regions. A new histogram for each new region is used to subdivide further. This is a recursive process and the algorithm requires input of a large number (17) of individual parameters. These parameters are what we wanted to optimize with a genetic algorithm. The implementation involved the use of much of the software previously developed in Sapphire including an evolutionary algorithms package and various other data structure templates. Although the system was not completed, we are expecting to get significantly better results than if we were to just use some default input parameters. Most of the implementation details were decided upon in conjunction with Chandrika Kamath. The evolutionary algorithm package was developed by David Littau, and much of the data-structure template structure was developed by Nu Ai Tang and Erick Cantu-Paz.

Building a Cache Component for the Memory Wall Project

Michael King
University of Utah

Summary:

Memory speed is growing at a much slower rate than processor speed. As time passes, the difference between the two speeds increases, making good cache design increasingly important. Because of this gap in speed, programs running on a multiprocessor system do not always make use of the high-processing rate potential. Our goal is to explore ways to optimize the use of cache memory in a multiprocessor environment.

Our approach is to build a trace-driven simulation of a memory system under representative applications in order to find out what optimizations to try on a real machine. Since we are not certain of every detail of the system we are simulating, it must be reconfigurable. We need to be able to change the hardware components that go into the simulation, as well as know how each component links to the others.

While in residence, I have built a reconfigurable cache component that simulates a generic cache. The simulation takes memory addresses as input and outputs statistics such as the hit/miss ratio and timing information. This tool should be useful in evaluating reconfigurable cache designs in the ongoing LDRD Memory Wall Project.

Sensitivity Analysis Techniques for Radiative Transfer

Rachel Knop
West Point Academy

Summary:

Radiation diffusion problems can often be modeled by a system of parameter-dependent differential equations. In many cases, it is not only important to solve the differential equations but also to quantify the sensitivity of the results to the problem parameters. These sensitivities can then be used to determine which parameters are most influential in affecting the simulation results.

The goal of this project was to compare methods for computing the sensitivity of a radiation diffusion problem with respect to a specific parameter. The model was comprised of a coupled set of differential equations for radiation energy and material energy. The model was numerically solved using PVODE, an ordinary differential equation solver. The parameter we considered was actually a scalar value used to multiply the Planck opacity values that appear in the coupled equations. The nominal value for this multiplier was 1.0. The so-called "brute force" method computed Planck sensitivity by taking the difference between the nominal results and results obtained when the multiplier was perturbed slightly, and then dividing these differences by the value of the perturbation. The magnitude of the perturbations ranged from 0.1 down to 0.000001. These sensitivity results were then compared with those obtained using SensPVODE, a sensitivity analysis version of PVODE. The SensPVODE approach involved deriving a differential equation for the Planck sensitivity and then simultaneously solving the radiation energy, material energy, and Planck sensitivity differential equations. With full error control, SensPVODE made step size and order selections based on the behavior of this entire system of differential equations. With partial error control, SensPVODE excluded the Planck sensitivities from the step size and order decisions.

We compared the brute force sensitivity method with both versions of SensPVODE (full error control, partial error control). The three sensitivity methods computed the Planck sensitivity at the same set of output times. Differences between Planck sensitivities were measured in terms of maximum relative differences, with the relevant SensPVODE results used as the reference solution. Initial results indicate that the differences were comparable when the brute force method was compared to the full or partial SensPVODE method. These sensitivity differences were relatively small initially (10% at most), and then again at later times when the radiation energy and material energy approached steady state. The most significant differences occurred at certain output times, roughly one-quarter to one-half of the way through the simulation. The brute force method is appealing because it computes sensitivities at far less cost than either SensPVODE method. The main objective in continuing this study will be to determine whether perturbation values that give brute force sensitivities as accurately as either of the SensPVODE methods can be found.

Parallel Implementation of an Algebraic Mortar Method

Tzanio Kolev
Texas A&M University

Summary:

We developed a parallel code implementing the mortar method with algebraically constructed multiplier spaces. The target application of this code was the construction of parallel multilevel preconditioners using element based (AMGe) coarsening in each subdomain. The work was carried out in close collaboration with Professor Joseph Pasciak from Texas A&M University and Dr. Panayot Vassilevski from CASC at LLNL. Parallel methods for numerical solution of elliptic partial differential equations typically require decomposition of the original computational domain into subdomains that are assigned to individual processors. Generally, the subdomain triangulations (meshes) may not align across the interfaces. The mortar method is a technique that allows for non-matching discretizations by imposing continuity of the finite element solution across the interfaces in a weak sense. More specifically, the jump of the discrete solution on each interface is orthogonal to a mortar multiplier space.

In this project the algebraic mortar technique (a pure algebraic extension of the finite element mortar method) was used to handle the case of parallel element coarsening with application to AMGe. The element based algebraic multigrid (AMGe) is an algorithm to coarsen generally unstructured finite element meshes where only the fine-grid element stiffness matrices are given. The AMGe operates on and produces generalized "elements" defined in terms of relation tables between nodes, faces, and elements. Even if the initial mesh is matching, independent (parallel) AMGe coarsening in each subdomain will produce non-matching meshes. These non-matching meshes can be "glued" by a mortar space to form a global problem. We considered an algebraic extension of the local construction for the mortar multipliers based on the general 3D dual finite element basis described by Kim, Lazarov, Pasciak, and Vassilevski.

A general code was implemented to illustrate the behavior of the proposed method. It requires input data for each subdomain that includes the element topology, the local subdomain stiffness matrices, as well as the mass matrices on the interfaces. This information is independent of the dimension and structure of the problem and is regenerated after an AMGe coarsening.

A number of tests for 3D problems were performed. An initial, generally unstructured tetrahedral mesh created by the mesh generator NetGen was pre-refined, then partitioned with METIS and then refined (now locally) independently in each subdomain to produce a generally non-matching mesh for the whole domain. The refinement was performed by ParaGrid, a parallel refinement code available in LLNL. A visualization tool based on OpenGL that allows for three-dimensional exploration of the solution was also developed.

The implementation constructs the initial mesh and calls an external AMGe coarsening subroutine developed by Dr. Panayot Vassilevski. The communication on the interfaces was implemented in an abstract way that can run on serial machines as well as in parallel based on MPI. The main target application of the code, a parallel multigrid preconditioner for the mortar system on the finest level, is being tested.

The Generation of Optimizing Preprocessors Using the ROSE Infrastructure

Markus Kowarschik
University of Erlangen-Nuernberg, Germany

Summary:

Large application codes in scientific computing are usually based on the use of libraries providing various data structures and functions. This is particularly true for object-oriented applications where these libraries implement high-level abstractions like array classes, grid classes, etc. Unfortunately, the use of high-level abstractions, which are defined in underlying libraries, cannot be optimized by the compiler since the semantics of these abstractions are user-defined and, as a consequence, unknown to the compiler. This lack of knowledge is a major reason for the poor efficiency of many object-oriented scientific codes: the Mflop/s rates that can be measured at runtime are just tiny fractions of the theoretically available peak performances that the manufacturers of the machines claim for their products.

ROSE is a software infrastructure for generating library-specific preprocessors that perform source-to-source transformations, e.g., eliminating the need for creating temporary objects and introducing cache-based transformations into computations on structured grids. Internally ROSE parses the application code (currently C++ code) and assembles the corresponding abstract syntax tree (AST). The use of library-specific high-level abstractions is recognized using high-level grammars, which are derived from the grammar of the base language C++. Since the application might be based on several libraries there might be a whole hierarchy of ASTs. After the ASTs have been modified according to the transformations specified by the library writer(s), the base language AST is unparsed. This final step yields the optimized C++ source code.

The work during this summer has mainly focused on two issues. First, we have investigated techniques for the specification of the source-to-source transformations. We have implemented a specification mechanism that is based on the assembly of optimized C++ code using strings of source code. This source code is then passed to the C++ compiler front-end, which generates the AST fragment to be plugged into the application's AST replacing the original AST fragment to be optimized. Second, we have implemented the automatic generation of a general-purpose tree traversal mechanism, which can be used in various places within ROSE, e.g., in order to simplify querying the AST during the transformation phase and in the course of unparsing the C++ AST after all transformations have been applied.

Parallel Unstructured Adaptive Multigrid for Instationary Problems

Stefan Lang
University of Heidelberg

Summary:

High-level parallel applications based on the message-passing paradigm are difficult to design and implement. This is especially true when solution adaptive techniques are used and problems on complex geometries are faced. Many of these difficulties are addressed inside the UG (Unstructured Grids) platform, e.g., dynamic load migration and load balancing, parallel grid adaption, and parallel I/0 and graphics. The driving idea during the design process of UG, a multigrid code for the computation of partial differential equations, was to find proper abstractions for each of the different functional parts of a parallel, adaptive, and unstructured software framework. This assures a maximal degree of code reuse and treatment of various partial differential equations becomes possible without superfluous coding effort.

Here we describe the usage of the UG library to compute density-driven flow through a ground layering system consisting of 63 different geological layers with permeability values changing by a factor of 10**4.

Time discretization is done with a BDF method. Space discretization is done by an implicit finite volume scheme. Using Newton linearization the nonlinear system is transformed into large sparse linear systems. These are then solved by parallel multigrid with BiCGSTAB. For visualization purposes, a scalar output, the solution in terms of the salt mass fraction is drawn using the parallel graphics device based on distributed z-buffers.

Using 1, 4, 16, 64, 256 and 512 processors in a scaled-sized computation, we computed up to 60 million unknowns. The linear solution process is analyzed for numerical and parallel speedup (efficiency). We show an average increase in number of iterations from 35.5 to 51.8, leading to a numerical efficiency of 0.61. Iteration time increases from 4.2 on 1 processor to 6.9 seconds, leading to a parallel efficiency of also 0.61. Thus, the overall efficiency is 0.61*0.61 = 0.37 and the corresponding speedup is 190. Timings for visualization range from 3.6 on 1 processor up to 4.6 seconds on 512 processors for creating one frame for the final movie.

In the future we will undertake parallel 3D calculations on processor numbers exceeding 10**3 for a variety of timesteps. We will perform scalability analysis of the linear and nonlinear solution process. We expect to be able to analyze up to 10**8 unknowns, a commonly cited ASCI goal, examining especially grid convergence behavior at sensitive points of the computational domain.

Data Classification Using Decision Trees

David W. Littau
University of Minnesota

Summary:

Decision trees are used to classify data. They use a set of labeled data to train, or build, the decision tree, based on the values of the data attributes. There were two kinds of decision trees typically considered: Axis-Parallel (AP) and oblique. An AP tree splits the data based on one numeric attribute, and an oblique tree splits the data using a linear combination of all the numeric attributes. A numeric split results in two new branches. If a given attribute is non-numeric, it is split into one branch for each separate nominal value. The tree construction uses a greedy algorithm that minimizes the local error for each split. Once the tree is built, it can be used to classify all the unlabeled data associated with the labeled data.

The Sapphire library already contained a fully operational implementation of the AP decision tree code. The capabilities of the library were expanded by adding code that created approximate AP trees, oblique decision trees using the OC1 algorithm, an approximate OC1 implementation, and oblique decision trees using an Evolutionary Algorithm (EA). The approximate AP trees differ from standard AP trees in that the data are not sorted to build an approximate AP tree. Instead, a histogram is created and the histogram boundaries are used to calculate split points. The OC1 code implements a standard oblique decision tree algorithm that uses an AP split as a starting point. The approximate OC1 code uses an approximate AP split as a starting point, as does the EA code. The oblique EA code also required some modifications in and enhancements to the existing EA library code. Any desired EA, be it a Genetic Algorithm, an Evolution Strategy, Evolutionary Programming, or a hybrid of all three types, can be used to evolve the oblique splits.

Experiments were performed using the approximate AP trees. An enhancement was added that randomized the split point for each branching, and an ensemble of 10 of these randomized trees was trained. The ensemble classified the data using a majority vote. The results for a variety of data sets indicated that the ensemble was, in most cases, better at classifying the data than a single regular AP tree. More experiments are planned by the Sapphire group, which will result in at least one publication. The new decision tree code will also be used as part of the ongoing effort in Sapphire to classify scientific data sets.

Automatic Performance Experiment Management and Analysis

Michael McCracken
University of California, San Diego

Summary:

Performance experiments are a common task for researchers at LLNL. The experiment space for these can be characterized by a set of control variables such as number of processors or an environment setting. Usually, there is a small set of metrics in which the experimenter is interested, for example, application runtime or cache utilization. The growing scale of parallel applications and the customizability of runtime systems can result in a very large experiment space that might be interesting to a researcher. The usual way to tackle such an experiment is to change settings, recompile, and run by hand, or to write a custom script that takes care of running the experimental trials but probably cannot assemble the data. This approach is labor intensive and a waste of the researcher's time. Because of the changing nature of the Laboratory's computing systems, a way to automate this common task is required that does not limit the scope of experiments that can be performed.

We have developed a tool, called Sergeant, that is flexible and general enough to manage complicated performance experiments and powerful enough to assist in interpreting the data from these experiments. It allows the description of the experiment space in a few short lines of a configuration file, and relieves the experimenter from the burden of rewriting the program logic that controls the execution of each experiment. The same compact file can describe the data that can be gathered from the experiment, and it takes the work of collating the data away from the experimenter, who is then able to browse only the results that are deemed interesting. Actually running experiments with the Sergeant tool is completely automatic, freeing the researcher to pursue other tasks. Several control space search strategies were investigated to intelligently limit the number of experiments run and to trim the amount of data produced. These strategies included hill-climbing optimization, factorial experimental design and analysis, and correlation analysis.

Use of Sergeant for performance experiments results in a significant decrease in the amount of time it takes to plan and begin such an experiment, a reduction in the amount of redundant programming a researcher must do. It lowers the amount of data that the researcher must search through to make useful conclusions about the experiment. A graphical interface for performance data visualization was developed for Sergeant, which makes the process of analyzing experiment results and producing graphs faster and much more conveniently.

Further development of Sergeant is planned, including investigation of more advanced search strategies and intelligent exploration of the experiment space.

Dynamic Detection of Streams in Memory References

Tushar Mohan
University of Utah

Summary:

With processor speeds doubling every eighteen months and main memory latencies reducing at only seven percent annually, application performance is becoming increasingly memory bound. Computer architects attempt to hide high main memory latencies by adopting techniques such as prefetching and using multi-level cache hierarchies. An application's performance on such architectures depends on the actual memory access patterns exhibited. The ability to recognize streams-sequences of evenly spaced addresses makes it possible to exploit knowledge of future accesses to improve memory system efficiency, as in smart prefetching memory controllers and parallel vector access mechanisms. Earlier work characterized an application's reference patterns using static, compile-time analysis. We present an efficient on-line algorithm to detect streams at run-time by analyzing the memory references issued by the program. By dynamically detecting streams, we avoid the need to create large trace file—issue for all but very short runs of trivial programs. We have implemented our algorithm and have used it for detecting streams in some common codes like matrix multiplication and DAXPY.

Our algorithm detects streams, which we define as a sequence of evenly spaced references. Other stream definitions are possible, but we choose a linear definition because of the widespread occurrence of such streams in real applications. Our algorithm detects streams interleaved with other streams and with irregular references, as well. It does this by computing the stride for a new reference with respect to previous references. On finding a sequence of equal strides greater than a minimum length, the sequence is classified as a stream. We limit the spatial and computational complexity of our algorithm by aging accesses. Older references that are not recognized as part of any stream are discarded as new references are added. We implement this by maintaining a window of active references; thus, streams whose references are further apart than the window size will not be detected. Our algorithm maintains a stream table of detected streams, each of which is succinctly represented as a tuple: starting reference, length, and stride. These tuples are conveniently stored in a chained hash with the expected stream successor as a hash key.

New references are stored in a pool until they can be classified into a stream or discarded through aging. Detecting streams requires determining the differences between a reference and its predecessors in the pool. To reduce the complexity of recomputing these differences as new references are added, we extend the pool to make room for differences to be stored along with the reference. As references age, their differences are retired as well.

The algorithm proceeds by appending new references to the pool. A hash lookup is then performed on the stream table to determine whether the new reference extends an already detected stream. If so, the stream table information is updated and the pool slot is marked to prevent difference calculations for the reference, as well as to ensure its exclusion in subsequent stream detection. If the reference does not extend any stream, its differences with its predecessors in the pool are computed. The differences are then scanned for streams. If a stream is detected, its accesses are removed from the pool and the stream is inserted into the stream table.

Summary (continued):

Tushar Mohan
University of Utah

We have implemented our algorithm and run it on commonly used codes. Though we have used a feature of the compiler on SRC-6 to instrument source codes presently, we are also developing novel techniques for dynamic binary instrumentation to record data references without the need for any compiler support. Currently, the compiler inserts a subroutine call before every load and store, with the address and the type of memory reference passed as parameters to this subroutine. Our algorithm is run as a separate process, and IPC is accomplished through named pipes.

We applied our stream detection algorithm to a few regular codes, like iterative matrix multiplication and DAXPY. We found that our algorithm correctly identifies streams occurring in the references. In matrix multiplication we found the absence of some expected streams and the detection of other streams of longer length. We will be applying our algorithm to a wider variety of codes in the future.

Large-Eddy Simulation and Multigrid Methods

Sandra Naegele
University of Heidelberg

Summary:

Our objective is the use of multigrid solvers for turbulence calculations using Large-Eddy Simulation (LES) modeling. Both multigrid and LES are based on multiscale phenomena and therefore multigrid methods seem well suited for LES calculations. There are some aspects that have to be considered, like the coarse-grid operator, transfer operators, and parallelization, when LES is used in conjunction with multigrid as solver.

The incompressible Navier–Stokes equations are discretized by a finite volume method of second order, and to discretize the system in time a implicit Runge–Kutta Method of second order is used. The unresolved scales of motion, the subgrid scales, are modeled by eddy viscosity type models or mixed models where for both types a dynamic determination of the model parameter is applied. The computational grid is adaptively refined to increase the resolution and decrease the model effort in areas of the domain where complicated structures are located. For the parallelization of the smoother a block-Jacobi approach is applied and the load balance itself is done via recursive coordinate bisection (RCB).

Some simulations were run on the ASCI Blue machine which gave fruitful experience in parallel calculations with regard to Jacobi-effects for the smoother on coarse grids and also with respect to parallelizing the turbulence model.

The main topic will be to use adaptive refinement, and due to that, very unstructured grids for 3D test problems like the flow around a cylinder. The cylinder flow is a very complicated problem where the necessary resolution differs strongly throughout the computational domain. Furthermore, the behavior of the solver as well as different approaches to construct the coarse-grid operator will be investigated.

Modeling Chemical Kinetics in Turbulent Combustion Simulations Using Overture

Diem-Phuong Nguyen
University of Utah

Summary:

The research objective of reaction modeling is to computationally link large, chemical kinetic mechanisms to turbulent combustion computations. The link must incorporate small-scale chemistry into large-scale components of the turbulent flow and systematically reduce the degrees of freedom of the system. The bridging between microscopic details to macroscopic domain is achieved through introduction of a subgrid scale (sgs) reaction model. Thus, my summer research at LLNL involved using the Overture framework to incorporate an sgs reaction model to the OverBlown Navier–Stokes solver to simulate a turbulent open pool fire.

The open pool fire simulation consists of calculating a transient turbulent flow coupled to gas phase kinetics and soot particle dynamics. The OverBlown solver will be used to solve the compressible Navier–Stokes equations along with an additional scalar transport equation for mixture fraction. The CFD is coupled to the reaction kinetics via the mixture fraction and enthalpy. The reaction model then provides the CFD with updated values of the local density, temperature, rates, and state space.

For the open pool fire, the traction free boundary conditions developed by Boersma will be implemented. These conditions require that all domain boundaries except for the outlet adhere to a Neumann condition where the derivatives are set to zero. The outlet adheres to a specified convective boundary equation.

Regarding the fluid dynamics, the Smagorinsky LES turbulence model will be implemented. Molecular diffusion will be neglected and the turbulent diffusion coefficient will be modeled as the turbulent Schmidt number times the momentum diffusivity obtained from the Smagorinsky constant.

In terms of reaction kinetics, two different models are available for coupling to the CFD. They are the equilibrium model and the intrinsic lower dimensional manifold (ILDM) method. Equilibrium consists of gas phase species only while ILDM captures non-equilibrium behavior and incorporates data from 800 reactions and 185 species including soot.

Interface code was written to couple the reaction model module with the OverBlown cfd. The ILDM implementation required an additional interface with a KDTree lookup. Bill Henshaw of LLNL is currently adding LES turbulence modeling, Boersma boundary condition, and scalar transport capability to the OverBlown CFD. Thus, the framework is in place for a series of simulations. This current framework is highly flexible and different reaction models, reactants, and fuels may be used.

I will run the open pool fire simulations and validate the data this fall. I plan to use Overture and OverBlown as the CFD platform to test out different reaction models. I would also like to use the complex geometry capability of Overture to apply turbulent reacting combustion simulations to industrial applications such as reactors. Finally, I would like to investigate the system in a parallel environment.

Space-Time Multigrid

Luke Olson
University of Colorado at Boulder

Summary:

Multigrid methods are considered well-developed and very efficient for elliptic boundary value problems and, depending on the discretization, multigrid methods can also be tailored for non-elliptic problems. In time-dependent problems, the use of semi-discretizations and time-stepping methods, which often result in elliptic problems at each step, are the method of choice. There are potential drawbacks in this framework, however. In large problems, parallelism becomes desirable, yet scalable time-stepping methods are rare and complex since they are inherently sequential in design. Furthermore, adaptivity and local mesh refinement are essential in many applications where a small part of the domain needs higher resolution while other parts of the domain need only coarse grids. In time-dependent problems, the areas in need of refinement not only exist spatially, but also in time. Adaptivity and some amount of parallelism are obtainable, but there is a cost involved. Part of the summer research project was to investigate these costs. Another goal was the study of the types of time-dependent problems that might benefit from a space-time multilevel approach and the discretizations that set up a convenient framework for efficient multigrid algorithms and adaptivity for these applications.

The software package *hypre* (High Performance Preconditioners) provided the necessary tools to solve these sparse linear systems in parallel. The flexibility of *hypre* enabled us to compare the "general" cost of solving a problem in space-time with multigrid versus stepping through time through SMG (semi-coarsening multigrid, a.k.a. Schaffer multigrid), a robust multigrid method for anisotropic arid variable coefficient problems that takes advantage of operator-based interpolation, line-relaxation, and semi-coarsening.

The FOSLS (first-order system least squares) methodology seeks to meet the goals of speed and adaptivity. FOSLS provides a sharp error estimator and in many cases leads to a form for fast multigrid convergence. Also, the growing interest from neutron transport practitioners indicates value in future research in this area.

FOSPACK Programming for Elasticity

Moongyu Park
Purdue University

Summary:

The standard Galerkin procedures for elasticity have many practical difficulties (e.g., the “rocking phenomenon”) as the material tends to become incompressible, i.e., the Poisson ratio tends to 0.5. Many people have attempted to develop alternate approaches that can overcome the difficulties in the incompressible limit. However, these approaches are usually based on mixed formulations that lead to discrete equations that are difficult to solve. Cai, Manteuffel, McCormick, and others developed first-order system least-squares (FOSLS) approaches for which Ruge wrote an implementation (FOSPACK) using algebraic multigrid methods (AMG), which is the main tool of our investigation.

We have tested the accuracy and convergence factor of FOSPACK for elasticity problems on several domains such as square, L-shape, horseshoe shape, rectangular domain with a rectangular hole, circular domain with a circular hole, and a rectangular domain with a circular hole. It has O(h**2) accuracy on square domains and a little lower accuracy on other domains because of the geometry irregularities of the domains.

With these successes demonstrated, we will be working on other model problems and developing more robust and higher performance implementation.

Measuring the Regularity of Array References

Erin Parker
Purdue University

Summary:

The running times of large scientific programs are strongly influenced by the time spent accessing main memory. Many mechanisms, such as prefetching, exploit regular access patterns in order to overlap memory accesses with computation and, thus, reduce memory stall cycles. The benefit of these mechanisms depends on the regularity of an application's memory accesses. Although several access descriptors have been proposed, access regularity is an intuitive concept for which few formal metrics exist.

We consider a program to be regular if it contains array references with identifiable access patterns that are repeated as memory is traversed. For our purposes, we restrict this definition to linear patterns. We present a set of metrics that quantify access regularity. We have implemented a source-to-source compiler mechanism to measure access regularity. Results on our sample code demonstrate that our analysis mechanism is fast and accurate.

We present three approaches for measuring the regularity of a program. Our static approach is a low run-time overhead mechanism that uses statically determined information, augmented at run time only by simple scalar data, such as loop bounds. Our dynamic approach instruments array references so that their regularity can be precisely determined at run time; this approach has significant run-time overhead but is highly accurate. Our overall approach is a hybrid of the static and dynamic approaches. It provides high accuracy with reasonable run-time overhead by using statically determined information where possible.

The static approach examines a program's AST (Abstract Syntax Tree) at compile time to gather knowledge of its loop nests and the array references made within the loop nests. Based on analysis of the array index expressions, we categorize an array reference as regular, irregular, or indeterminate. A regular array reference is one in which all indices are linear expressions of the LCVs (Loop Control Variables). An irregular array reference is one in which at least one index is a nonlinear expression of the LCVs. An array reference is indeterminate if at least one index is an expression that cannot be analyzed at compile time or the array reference is contained in the body of a conditional. For example, the array reference A[B[i]] is indeterminate without knowledge of how array B is initialized, and the array reference A[f(i)] is indeterminate without knowledge of what is returned by the function f given input i. Although more aggressive compile time analysis can categorize some occurrences of these two examples as regular or irregular array references, in general, their regularity cannot be determined until run time.

For any regular array reference, each execution of the innermost loop enclosing it will generate a predictable stream of array accesses. We call such a stream a regular stream. Our linear restriction implies that array references we classify as irregular do not constitute a regular stream. Indeterminate array references may be irregular; our static approach assumes that they are. Therefore, based on analysis of the LCVs of the loop nests containing regular array references, we can compute the number of regular streams, their average length, and the proportion of array accesses that occur in regular streams, among other statistics.

Summary (contiued):

Erin Parker
Purdue University

The dynamic approach examines a program's AST to locate array references contained in loop nests. It does not analyze the indices of array references or LCVs of loop nests. Instead, we instrument the AST with instructions for tracking the actual value of the index to an array reference. A stream of indices forms a regular stream if the stride between all values is the same. We keep the same statistics for regular streams as in the static approach. The dynamic approach accurately categorizes all array references although it makes no attempt to categorize them statically.

The hybrid approach combines the two approaches described above. As in the static approach, we categorize an array reference as regular, irregular, or indeterminate. Then for every regular array reference, we compute the statistics for its regular streams. However, instead of conservatively assuming that every indeterminate array reference is irregular, we perform run-time tracking of array indices to discern actual regularity, as in the dynamic approach.

It is clear that the static approach incurs virtually no run-time overhead, but its accuracy can vary widely and is based on the number of indeterminate array references in a program. The dynamic approach enjoys great accuracy at the cost of a noticeable run-time overhead. The hybrid approach is designed to incur larger run-time overhead only when it is necessary for greater accuracy.

We accomplish automatic analysis and instrumentation of the AST using ROSE. ROSE is a tool for building source-to-source preprocessors. The preprocessor generates an AST from the program source code; the AST is then used for analysis, instrumentation, or optimization. The instrumentation of our static approach merely computes the regularity statistics once the values of any run-time constants are known. In our dynamic approach, our instrumentation actually tracks array index values and detects any regularity. The hybrid approach only uses the more expensive run-time instrumentation for indeterminate references.

We discuss the accuracy and run-time overhead of our three approaches for a simple test program. This example program clearly demonstrates the trade-offs between our approaches.

```
do i = 0, regularity_param

do j = 0, MAX

sum += A[j]

do i = 0, 100 - regularity_param

do j = 0, MAX

sum += A[B[j]]
```

Note that regularity_param is an integer provided by the user at run time whose value is between 0 and 100. B is an array of integers with size at least MAX, which has been initialized in one of two ways. In Case 1, B[i] is a random integer with a value between 0 and MAX-1, and in Case 2, B[i] = i.

Summary (contiued):

Erin Parker
Purdue University

Our example program has three array access streams: the accesses to the A array in both loops and the accesses to the B array that determine the A indices in the second loop. All three of our approaches correctly detect regularity in Case 1 of the sample program. However, for Case 2 of the sample program, our static approach misclassifies array reference A[B[j]] as irregular, while our dynamic and hybrid approaches correctly classify it as regular.

The running time required by the source code instrumented using our hybrid approach is proportional to the number of indeterminate array references that must be tracked at run time, as expected. The run-time overhead of our hybrid approach is significantly less than that of our dynamic approach even when the value of regularity param is 0. Although the indeterminate array reference A[B[j]] must be instrumented for run-time detection of regularity, our hybrid approach saves run-time overhead by statically categorizing the array reference B[j] as regular.

The effort to measure regularity in programs is ongoing, and the preliminary work discussed here has raised several issues. It is undesirable to use our dynamic approach to measure the regularity of large LLNL codes, as it will add overhead to already long-running programs. Likewise, the possible inaccuracy of our static approach on complicated programs makes it unsuitable. Therefore, we are interested to see the accuracy/overhead trade-off of using our hybrid approach on such programs. Furthermore, our analysis can be expanded to include references to array class objects in use at LLNL, which we expect to introduce new challenges.

A Shrink-Wrapping Approach to Large-Scale Visualization

Serban Porumbescu
University of California, Davis

Summary:

Scientists are running simulations that generate data sets several orders of magnitude larger than what current technology can handle. Our goal is to create algorithms that allow interactive manipulation and visualization of these large data sets.

Our technique, called "shrink wrapping," takes a generated isosurface (mesh) and reparameterizes it so that wavelet compression can be applied. This summer we modified our current approach by eliminating the construction and use of the signed distance field due to surface folding problems we could not resolve. Our new approach creates a lower resolution mesh by performing a topology preserving decimation of the initial isosurface. This newly created mesh is then also decimated to create a third mesh. The process continues until we generate a base mesh with a suitably low resolution. At this point we have a continuum of meshes where each mesh differs from its finer and lower resolution version by a very small amount. Beginning with the base mesh, we take each mesh in turn and perform a series of subdividing, snapping, and smoothing operations in relation to the mesh at the next finer level. The idea is to capture important features by gradually introducing them as we move from coarse to fine meshes.

Our future work entails fine-tuning the newly developed technique, parallelizing our algorithm, and applying an adaptive mesh approach that performs the subdivision operation only in areas of high detail.

Simulation Tool For Studying Solutions to the Memory Wall

Pete Poulos
University of Utah

Summary:

Processors are getting much faster than memory systems. These diverging speed growth curves create a memory bottleneck in almost all computer systems, but the problem is particularly acute in symmetric multiprocessor systems (SMPs). The Memory Wall project is developing techniques to improve the performance of the memory hierarchy in SMP systems. This requires exploring design alternatives for smart memory controllers. To this end, we are developing a simulator that will allow us to gather statistics about the performance of the various design alternatives without having to build each candidate in hardware. This will save time and money, as it is easier to simulate a system than it is to build one.

We need to be able to simulate a wide variety of hardware configurations. Also, the simulator needs to run fast and simulate parallel architectures. The latter requirement distinguishes it from most publicly available simulation tools. In order to make the simulator efficient, we have designed it to be easily parallelizable into a number of threads to be run on a parallel computer. The parameters of the machine being modeled and the statistics to be gathered during simulated execution are settable at run-time via a single configuration file.

We are using run-time trace-driven simulation, where each input trace is gathered from a single node of a baseline SMP system. These memory traces are presented to our simulator, which calculates timing information for each access on the memory system being simulated. Using traces allows us to abstract away all details of the processor microarchitecture and instead focus on the memory behavior. This lets us model any processor and increases simulator efficiency, but potentially sacrifices accuracy in measurements. This is an appropriate tradeoff for this particular research project. This type of simulation is not publicly available.

Many of the individual system components have been developed, but we are still implementing the system-wide architecture. The API between the program being traced and the simulator is being developed, as well as the components that manage the division of components across different threads.

Gesture-Tracking for Visual Intuitive Navigation of Large Data Sets

Min C. Shin
University of South Florida

Summary:

We have been developing a new approach to a gesture-tracking system using real-time range on-demand. The system represents a gesture-controlled interface for interactive visual exploration of large data sets. The paper describes a method performing range processing only when necessary and where necessary. Range data is processed only for depth of interest. This is accomplished by a set of filters on the color, motion, and range data. The algorithm also includes a robust skin-color segmentation insensitive to illumination changes.

We have tracked the manipulating hand nearly at four frames per second. By using change of hand area, we have identified the opening (rapid increase of area) and closing (rapid decrease of area) signifying the beginning and ending of the gesture. Using the range images, we have computed the 3D trajectory of the manipulating hand. We fitted the 3D Bezier curve using the 3D trajectory. Then, the curvature and the trajectory direction are used to determine the gesture. If the curvature is high, the gesture is recognized as rotation. If the curvature is low indicating a linear motion, we have classified the gesture as zooming if the direction is mostly in z-axis; otherwise it is classified as translation.

We have tested the algorithm using 398 images of two people consisting of 16 gestures. The algorithm was able to detect the manipulating hand 99% correctly. The beginning and ending of gesture were detected 91% correctly. Overall, the gestures were recognized correctly with an 88% success rate.

We aim to further improve the speed by computing range on selected regions (range on demand). We will also connect the system to a visualization package to demonstrate the application of the gesture tracking.

Parallel Grid Refinement and a Posteriori Error Analysis

Stanimire Z. Tomov
Texas A&M University

Summary:

Parallel grid generation tools play an important role in scientific research. To enable the development of efficient computational technologies, such tools may have to generate finer meshes only in some regions of the computational domain. This could be achieved by applying a posteriori error analysis. The goal of my summer project was to work on this problem area. More precisely, my goal was: (l) to further develop a parallel grid generation tool for three-dimensional problems; (2) to help other researchers in CASC to use it for algorithm testing purposes; (3) to integrate it with *hypre* data structures; (4) to use it with *hypre* preconditioners through the Finite Element Interface (FEI); and (5) work on a posteriori error analysis from theoretical and practical points of view.

A parallel mesh generation tool, named ParaGrid, was further developed. The new development was a continuation of a two-dimensional project begun last summer. ParaGrid is software that takes as input a coarse tetrahedral mesh, which describes well the domain, splits it using the METIS partitioning software, distributes the partitions among the available processors, and generates in parallel a sequence of meshes. It has internal solvers and is able to generate various Finite Element/Finite Volume discretizations. The data structures allow ParaGrid to be easily connected to (or used to provide data to) external parallel finite element/volume solvers based on domain decomposition. It has been successfully used from several researchers in CASC for algorithm testing purposes.

I worked with Charles Tong on data structures for parallel finite element problems. The stress was on the generation of a parallel element topological data structure. I finished the generation of *hypre* matrices by providing the relations "elementnode," "elementface," "facenode," and their transposes. The test data was generated with ParaGrid.

Generation and solution routines for elasticity problems were added to the code. *hypre* preconditioners and solvers can be used. The connection is done through FEI 3.0.

I completed with Dr. Raytcho Lazarov a study on a posteriori error control strategies for finite volume/element approximations of second order elliptic differential equations. These refinement techniques were applied to finite volume discretizations of various boundary value problens for steady-state convection-diffusion-reaction equations in two and three dimensions. The results were summarized in an article on "A posteriori error estimates for finite volume element approximations of convection-diffusion-reaction equations," which has been submitted to *Comput. Geosciences*.

An accompanying visualization a tool named GLVis was also developed. GLVis started from a 2D visualizer developed in a team project at Texas A&M University. Its features include solution visualization in moving cutting planes, input from files and AFJNET sockets, visualization of vector field, and displacements.

My work on error control, adaptive grid refinement, and a posteriori error analysis continues. In addition, mesh de-refinement software is under development for the case of adaptive grid refinement for time-dependent problems.

Input/Output Scalability of Different Architectures

Preethy Vaidyanathan
University of California, Santa Cruz

Summary:

We seek to understand how to develop file systems for low-cost cluster computers for biological applications. Many studies have been carried out with file systems for classical parallel applications (e.g., PVFS). Like other scientific computations, many computational biology applications are highly data parallel, but they have the distinct characteristic of being I/O intense.

We instrumented an application vital to the Human Genome Project, which consumes more than half of the execution time at the UCSC computational biology cluster using the Pablo Instrumentation library from the University of Illinois. We characterized the I/O behavior of this application on Linux clusters with different file systems (the UCSC cluster and the Vivid cluster) and an IBM SP/2 (ASCI Pacific Blue). Our study shows that disk-processor locality is a very important factor affecting the performance of this application on the cluster.

We have presented the design of a user-level library for a new model of location-transparent storage to automatically redirect read accesses to the most appropriate location for obtaining the best performance.

The Discretization of the Scattering Kernel with Angular Finite Elements

Nicolas Vallete
Texas A&M University

Summary:

The angular variables in the transport equation are usually discretized either by a spherical harmonic expansion or by a quadrature rule (so-called "Discrete Ordinates" methods). Discrete Ordinate methods often yield unphysical results, which are called ray effects. The goal of my summer project was to reduce the ray effects by discretizing the sphere with finite elements. The discretisation of the transport equation by piecewise constant finite elements leads to an equation that can be solved with existing algorithms.

The goals for the summer were to code the finite element representation of the scattering kernel, determine the underlying symmetry of the scattering kernel, code the symmetries into the function, which calculates the scattering kernel, and to implement the previous program into a parallel transport code in order to compare this method with others.

The initial program generating the integrals of the spherical harmonics was written in Matlab. I completed this program with some other subroutines in order to generate cross sections tables. I also wrote another program in Matlab, which reduces the amount of storage of the scattering cross section by a factor of 50. Future plans for this method include adaptive refinement on the sphere.

Diffusion Synthetic Acceleration for Three-Dimensional Transport Equations

S. Van Criekingen
Northwestern University

Summary:

The linear Boltzmann transport equation (BTE) is an integro-differential equation arising in deterministic models of neutral and charged particle transport. Iterative methods are routinely used to solve large equation systems resulting from discretization of the BTE. The diffusion synthetic acceleration (DSA) has been proved to be very efficient in preconditioning the one-dimensional BTE. Our aim was to extend existing one-dimensional convergence proofs to three dimensions.

The discretizations consist respectively of a standard discrete ordinates collocation of the angular variable, and a Petrov–Galerkin finite element approximation in space. DSA requires a "consistent" discretization of a limiting diffusion approximation to the BTE. While for 1D slab geometry the consistently differenced diffusion problem is nonsingular, it has been shown that the consistently differenced 3D diffusion approximation is actually singular, although the DSA preconditioner itself remains nonsingular. We extended the 1D theoretical results in the asymptotic diffusion limit to 3D and submitted our work to *SIAM Journal of Numerical Analysis*.

Additional work is needed to address the DSA method in the context of the corner balance spatial discretization method, for which some 1D results have been obtained already.

Adaptive Mesh and Algorithmic Refinement Simulations for Multiscale Hydrodynamics

Sanith Wijesinghe
Massachusetts Institute of Technology

Summary:

LLNL has a multifaceted research effort focusing on numerical, algorithmic, and software issues related to the use of structured adaptive mesh and algorithmic refinement (AMAR) technology. A key benefit of AMAR is the capability to investigate multiscale and multiphysics problems within a single computational environment. This feature is key to the analysis of many computational physics applications (e.g., Richtmyer–Meshkov instability) important to LLNL and the ASCI program.

The AMAR scheme used in the current research consists of a second-order Godunov Euler solver and a Direct Simulation Monte Carlo (DSMC) particle solver.

The objectives of the current research are as follows:

1) Assess AMAR simulation through extensive testing of model problems.

2) Investigate methods to track fluid discontinuities and material interfaces and how they impact the results obtained from AMAR.

3) Scale AMAR to large systems of interest.

Test simulations for stationary shock waves have been conducted. The results from these simulations are currently being compared with theoretical results. Concentration-based adaptive gridding criteria have been incorporated to track material interfaces. AMAR simulations of a step concentration discontinuity have validated these criteria. A large-scale parallel simulation of the Richtmyer–Meshkov instability has been conducted on the IBM ASCI Blue.

Future plans include improvements to the load balancing routines to allow more efficient parallelization of the AMAR scheme. Longer Richtmyer–Meshkov simulations will also be conducted to investigate the time evolution of the material interface.

Multiscale Simulation Combining Direct Simulation Monte Carlo and Navier–Stokes Solvers

Yihao Zheng
University of California, Davis

Summary:

Numerical modeling of fluids is particularly challenging when the problem of interest spans length scales differing by orders of magnitude. Even mesh refinement is not sufficient when the smallest length scale approaches the microscopic regime since standard hydrodynamics is not accurate. In such cases a different simulation algorithm, which contains the appropriate microscopic physics, is required but only at the finest level of refinement; this methodology is known as Adaptive Mesh and Algorithm Refinement (AMAR).

These types of hybrids, combining microscopic and macroscopic algorithms, are still under development and many critical challenges exist, yet their utility is well recognized in the field of computational fluid mechanics. Our specific project involves the implementation of such a hybrid code combing Direct Simulation Monte Carlo (DSMC) for particle simulation and a Godunov-type Navier–Stokes solver for continuum modeling, using the Structured Adaptive Mesh Refinement Application Infrastructure (SAMRAI).

The hybrid code combines two simulation elements, one of which (DSMC) was already in place at the start of the summer. The other half is a Navier–Stokes solver, which is being developed as a generalization and extension of the Euler solver currently used by the AMAR hybrid code. The work this summer consisted of reading the supporting literature and learning the existing code. A stand-alone version of the three-dimensional, second-order Godunov solver for the Euler equations was written and is now being tested.

Future plan call for generalizing the solver to the Navier–Stokes equations and integrating the new solver into the existing AMAR code.